VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

GiD jako preprocesor a postprocesor pro konečněprvkový systém GEM
Interfacing the Preprocessor a Postprocessor GiD with GEM Finite Element Package

Bakalářská práce

2012 Martin Klapuch

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Ing. Martin Klapuch**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: GiD jako preprocesor a postprocesor pro konečněprvkový systém GEM

Interfacing the Preprocessor and Postprocessor GiD with GEM Finite Element Package

Zásady pro vypracování:

Seznamte se s GEM, systémem pro 3D matematické modelování na bázi metody konečných prvků, a stávajícím způsobem přípravy modelů (preprocessingem) a zpracování výsledků výpočtů (postprocessingem).

Nastudujte GiD, univerzální grafické rozhraní pro tvorbu modelů a vizualizaci výsledků numerických simulací.

V GiD vytvořte rozhraní k systému GEM, aby byl s to načítat a zpracovávat data GEM.

Rozhraní otestujte na existujícím matematickém modelu.

Seznam doporučené odborné literatury:

V. Kolář et. al.: FEM principy a praxe metody konečných prvků. Computer Press, 2001, ISBN 8072260219
R. Blaheta et al.: GEM – A Platform for Advanced Mathematical Geosimulations. In: Parallel Processing and Applied Mathematics, LNCS 6067/2010, Springer Verlag, 2010, pp. 266-275, ISBN-10 3-642-14389-X, ISBN-13 978-3-642-14389-2
Dokumentace a kódy systému GEM, viz také http://www.ugn.cas.cz/?l=en&a=&p=other/sw-gem/gem.php
Dokumentace GiD, viz také http://www.gidhome.com/

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **RNDr. Ondřej Jakl, CSc.**

Datum zadání: 19.11.2010

Datum odevzdání: 04.05.2012

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: 15. 8. 2012 .....

(podpis)

Rád bych na tomto místě poděkoval vedoucímu práce, RNDr. Ondřeji Jaklovi, CSc., za motivaci, připomínky, čas a trpělivost. Dále bych rád poděkoval Mgr. Alexeji Kolcunovi, CSc. za jeho čas a odborné připomínky.

Abstrakt

Tato práce se zabývá rozhraním k systému GiD umožňující načítat a zpracovávat data programu pro matematické modelování GEM. V práci jsou popsány oba programy, způsob předávání výsledků GEMu a možnosti jejich vizualizace v prostředí programu GiD. Dále jsou nastíněny další možnosti jeho využití jakožto preprocesor úloh pro řešič GEM.

Klíčová slova

Metoda konečných prvků, matematické modelování, postprocesor

Abstract

The present thesis deals with the interface to the GiD system that enables to read and to process the program data for mathematical GEM simulation. Both programs as well as the way of handing over GEM results and possibilities of their visualisation in the GiD program setting are described in the thesis. Other possibilities of its use such as a preprocessor of tasks for GRM solver are also sketched.

Key Words

Finite Element Method (FEM), numerical simulations, postprocessor

Obsah

1 Úvod ... 6
2 Terminologie ... 7
2.1 Preprocessing ... 7
2.2 Calculating module ... 7
2.3 Postprocessing ... 7
2.4 Konečněprvkový systém ... 7
3 Metoda konečných prvků ... 8
4 GEM ... 9
4.1 Struktura programu GEM ... 9
4.1.1 GEM – preprocessing ... 9
4.1.2 GEM – řešič ... 10
4.1.3 GEM – postprocessing ... 11
4.1.4 Alternativní schéma - Systém GEM3 ... 11
4.2 Shrnutí ... 12
5 GiD ... 13
5.1 Popis systému ... 13
5.2 Řešení problematiky ladění GiDu - preprocesing ... 13
5.2.1 Problem Type ... 13
5.2.2 Předávání dat výpočetnímu modulu ... 14
5.3 GiD - postprocesing ... 15
5.3.1 Formát souboru „msh“ pro postproces ... 16
5.3.2 Formát „res“ pro postproces ... 17
6 Postprocessing ... 20
6.1 Znázornění sítě ... 20
6.1.1 Výběr funkcí pro znázornění sítě ... 20
6.1.2 Výběr dat pro znázornění sítě ... 21
6.1.3 Shlukování sítí ... 21
6.2 Znázornění výsledků ... 22
6.2.1 Výběr funkcí pro znázornění výsledků ... 22
6.2.2 Výběr dat pro znázornění výsledků ... 22
6.2.3 Postup ... 23
6.2.4 Vizualizace výsledků v programu GiD ... 23
6.2.5 Nastavení některých zobrazení ... 24
7 Závěr ... 27
8 Použitá literatura ... 28
Seznam obrázků ... 29
Seznam příloh ... 29
9 Přílohy ... 30

1 Úvod

Potřeba namodelovat fyzikální jevy nejen v prostředí hlubinného dobývání vedla k vývoji numerických metod řešících soustavy parciálních diferenciálních rovnic, na které běžné analytické metody nestačí. Díky těmto metodám lze odhadnout, namodelovat a předvídat procesy v horninovém masivu. Jedním z mnoha systémů matematického modelování řešících problémy geomechaniky je software GEM. Nedílnou součástí modelování je také reprezentace výsledků, které řešením úloh vzniknou. Výsledky rozložení teplot, hydrauliky či mechanických veličin reprezentovány ať už skaláry, vektory či tenzory, jsou nejlépe vystihnuty a pochopeny jejich vizualizací. Pro tyto účely byla vytvořena řada grafických aplikací, mezi které lze zařadit také software GiD. Jelikož GiD není jen grafická aplikace, ale také ucelený aparát matematického modelování, zmíníme se také možnosti využít program jako preprocesor, případně také jako samotný řešič.

V práci popíšeme oba výše zmíněné programy, přičemž se zaměříme hlavně na předávání výsledků programu GiD a možnosti jejich vizualizací. Vycházet budeme z popisu současného stavu a dostupnosti jiných alternativ. Práce by tedy měla nabídnout jednu z možností zacelení slabých částí programu GEM, proto si popíšeme každý oddíl programu spolu s jeho exportem dat. Následně si popíšeme rozdělení programu GiD a prostředky, se kterými přistupuje program k importu výpočtů. Završením práce je pak názorný popis možností práce s těmito výsledky přímo v programu GiD. Vše si ukážeme na několika příkladech práce s konkrétními daty poskytnutými Ústavem geoniky AV ČR, v.v.i.

Výběr programovacího jazyka pro implementaci rozhraní je závislý na možnostech, které nabízí knihovna Gidpost – nástroj programu GiD. Knihovna Gidpost je rozšířením buďto jazyka C++, nebo jazyka Fortran. Protože je systém GEM vyvíjen v jazyce Fortran77, a většina jeho výstupních dat je v binární podobě, bude implementace zpracována v jazyce Fortran77.

2 Terminologie

2.1 Preprocessing

Preprocessingem můžeme nazvat metodu, při které program diskredituje problém (viz níže). Preprocessing slouží k vytvoření informací pro následné výpočty problému. Tyto informace jsou *mesh* (sítě), *material* (materiálu) a *conditions* (podmínek).

2.2 Calculating module

Výpočetní modul, nebo také řešič, je zpravidla část programu, která zpracovává informace předané preprocessingem. Za pomocí numerických metod je problém řešen, a výsledky tohoto řešení jsou předány ke zpracování poslednímu kroku a nalýzy problému – postprocessingu.

2.3 Postprocessing

Postprocessing je konečná fáze vyhodnocení problému, při níž jsou zpracovávány výpočty vizualizačními nástroji.

2.4 Konečněprvkový systém

Výpočetní systém, který pro zpracování vstupních dat využívá numerickou metodu – Metodu konečných prvků.

3 Metoda konečných prvků

V geomechanice, která se zabývá problémy mechaniky zemin a hornin, se využívá matematické modelování pro řešení konkrétního problému. Matematické modelování je tvořeno souborem vztahů, podmínek a metod, s jejichž pomocí lze dojít k přibližnému řešení problému nad ohraničeným prostorem. Při modelování se vychází ze základních bilančních zákonů (Zákon zachování hmotnosti a Zákon zachování hybnosti), konstitučních vztahů, tedy charakteristikou materiálu (např. Hookův zákon, Stavová rovnice), okrajových a počátečních podmínek, které představují hodnoty veličin řešené úlohy ohraničeného prostoru. Završením matematického modelování je poté soustava parciálních diferenciálních rovnic, jejichž řešení je pro složitější případy značně obtížné. Pomocí numerických metod lze však dospět k přibližným výsledkům. Mezi numerické metody, používané v geomechanice, patří Metoda sítí, která je založena na diskretizaci parciálních diferenciálních rovnic, Metoda charakteristik, která se zabývá řešením hyperbolických úloh pomocí vztahů na charakteristikách, Metoda hraničních prvků, spočívající v transformaci okrajového problému na hraniční integrální rovnice a Metoda konečných prvků. V geomechanice se nejběžněji používá k řešení numerických metod Metoda konečných prvků

Metoda konečných prvků (FEM - Finite Element Method), sloužící k řešení komplexních úloh, vznikla ve 40. l etech 20. s toletí. Problém byl rozdělen na soustavu podoblastí, jejíž podstatnou vlastností byla elasticita v uzlových bodech. V 50. letech byla rovnice tuhosti definována v maticovém tvaru vhodném pro její řešení na počítači.

Základem metody je tzv. diskretizace, tedy rozdělení objektu na části, které lze matematicky popsat, čímž dojde k nahrazení reálného tělesa s nekonečným množstvím bodů za model tělesa s konečným počtem bodů (uzlové body). Pro každý z těchto bodů je definováno pole posunů a je spočteno pole deformací a pole napětí. Funkce posunů se nahradí polynomem a zavedou se okrajové podmínky. Dalším krokem je výpočet soustavy lineárních algebraických rovnic, dále je výpočet deformací a napětí pro každý uzlový bod.

Na trhu je celá řada programů numerických simulací, které umožňují pomocí FEM metody řešit problémy napětí a deformací, jako např. ANSYS, CFX, CALFEM, COSMOS/M, FELyX, FEniCS, FreeFem, GETFEM, libmesh, NASTRAN, NEXIS, OOFEM, PETSc, RELAX aj.[3]. Mezi tyto řešiče lze zařadit také program GEM, který byl vyvinut na Ústavu geoniky AV ČR, v.v.i., kde je v současné době stále využíván.

4 GEM

Konečněprvkový software GEM slouží k matematickému 3D modelování a simulací v geovědách, především procesů v horninách vlivem hlubinného dobývání nerostných surovin, ale také budováním různých podzemních děl, např. úložišť odpadů. GEM řeší úlohy mechanických změn a také úlohy tepelných dějů. Umožňuje tedy pochopit rozvržení sil v horninách a napomáhá zvyšovat bezpečnost procesů těžby uhlí.

Software, vyvinutý v Ústavě geoniky AV ČR, obsahuje moduly, tedy řešiče konkrétních problematik. Níže si popíšeme části softwaru, kterými se budeme dále zaobírat.

4.1 Struktura programu GEM

Aby mohl být program pro řešené úlohy matematického modelování komplexní, musí obsahovat všechny fáze vývoje úlohy od preprocessingu, přes samotný výpočet řešení, až po vyjádření výsledků pomocí postprocessingu. Také struktura programu GEM obsahuje tyto fáze. Pro definování úlohy jsou zde moduly Tgrid a Smat, výpočetním jádrem programu je modul Sesol a výsledky se exportují do Matlabu.

4.1.1 GEM – preprocessing

Řešení každé úlohy začíná fází její definice. V této fázi se vytváří síť, zadávají materiály a definují počáteční podmínky.

Vytvoření sítě je operace, při které je navržena geometrie strukturované sítě, která je deformována s ohledem na sledovaný problém. Informace o této síti nese složky souřadnic uzlů sítě, kód buňky a číslo materiálu v buňkách.

4.1.1.1 Tgrid

Tgrid jako preprocesor pro danou úlohu slouží pro generování vstupních souborů. Má využití převážně při přípravě sítě pro konkrétní úlohu. Buňkou sítě může být buďto kvádr, nebo také čtyřstěn. Takto vytvořená strukturovaná síť se pro účely konkrétní úlohy může dále rozdělit na šest čtyřstěnů (Kuhnsova triangulace). Při specifikaci úlohy je důležitá také definice a distribuce materiálů v síti. Data jsou předávána z klávesnice přes příkazový řádek.

Preprocesor obsahuje tyto prvky:

- tgrid.for - preprocesor
- tgrid.mk - pro překladovou dávku
- wfv.for - pro zápis základních informací o dané úloze do souboru fv.g32
- lin.mki - pro překlad pro linux
- ipcg.for - pro vnitřní metodu sdružených gradientů s předpodmíněním

Soubory generovány částí TGRID (tgrid.for). Jejich výčet a funkce jsou:

- fbc.g32 - informace o posunutí
- fel.g32 - informace o buňce (index INDZ pro uzly, materiál, determinant čtyřstěnu, indexy buňky, souřadnice uzlů)
- fmat.g32 - informace o materiálu (číslo mat., elasticita, specifická hmotnost, pozice, izotropie)
- fr.g32 - rozložení síly v uzlu
- fbc.g32 - informace o posunutí uzlu

- fv.g32 - základní informace o úloze
  (jméno úlohy, datum, počet uzlů, stupeň volnosti, materiál)
- fx.g32 - souřadnice uzlu diskretizační sítě

4.1.1.2 Smat

Oddíl programu Smat slouží pro sestavení matice tuhosti, která vztahuje uzlové síly a uzlové posuny.

Obsahuje tyto prvky:

- smat.for - sestavení matice tuhosti
- smat.mk - pro nastavení architektury
- smat.mki - pro parametry překladu pro jednotlivé architektury
- geore.for - pro vytvoření soustavy rovnic respektující kinetické okrajové podmínky ve 3D
- init.for - pro inicializaci, generování souboru fmt.g32
- lin.mki - pro překlad pro Linux
- load.for - pro výpočet vektoru zatížení
- rfv.for - pro čtení souboru fv.g32
- stiff.for - pro sestavení matice tuhosti

Soubory generovány části SMAT (init.for, geore.for). Jejich výčet a funkce jsou:

- fmt.g32 - informace o materiálu
  (čte ze souboru *fmat.g32*)
- frw.g32 - soustava rovnic resp. kinematické okrajové podmínky ve 3D
  (čte ze souboru *fbc.g32*)
- fkbc.g32 - sestavuje matici tuhosti
  (čte ze souboru *fmat.g32*, *fel.g32* a *fmod.g32*)

4.1.2 GEM – řešič

4.1.2.1 Sesol

Sekvenční řešič soustavy lineárních rovnic je založený na metodě sdružených gradientů s předpodmíněním. Předpokladem je, že matice soustavy je pozitivně definitní, matice je symetrická. Vhodně se volí vektory posloupnosti (posloupnosti iterací, reziduí a sdružených vektorů), určující směry postupu mezi aproximacemi. K urychlení výpočtu je využíváno tzv. předpodmínění, které za pomocí předpodmiňující matice zlepšuje vlastnosti matice tuhosti. Pro urychlení výpočtů se v praxi používá také paralelizace předpodmínění s využitím multiprocesorových systémů.

Data, se kterými pracuje program, jsou uchovávána v operační paměti. Řešič obsahuje tyto prvky:

- sesol.for - řešič
- sesol.mk - pro nastavení architektury
- sesol.mki - pro parametry překladu pro jednotlivé architektury
- rssin.for - pro čtení souboru sesol.ini
- rmat.for - pro čtení souboru fkbc.g32
- rfv.for - pro čtení souboru fv.g32
- pcg.for a ipcg.for - metoda sdružených gradientů s předpodmíněním
- pcond.for - inicializace předpodmínění

- mxv.for - násobení matice a vektoru

Soubory generovány částí SESOL (pcg.for, geore.for). Jejich výčet a funkce jsou:

- fu.g32 - uložené řešení
  (Metoda sdružených gradientů s předpodmíněním, čte ze souboru *frbc.g32* – pravá strana matice tuhosti)

Soubory generovány částí STRESS. Jejich výčet a funkce jsou:

- fsts.g32 - uložené řešení výpočtu napětí a energie vycházející ze souborů fv.g32, fmat.g32, fel.g32 a fu.g32

  Napětí v čtyřstěnu popisují tyto parametry:
  - IT(4) - globální čísla vrcholu
  - MT - číslo materiálu trojúhelníka
  - DT - souřadnicový determinant
  - ST(6) - vektor napětí
  - DF(6) - vektor deformace
- fmt.g32 - informace o materiálu

4.1.2.2 Iterační řešič ISOL

Paralelní iterační řešič ISOL je volný software určený pro řešení rozsáhlých lineárních systémů. Algoritmy, podle kterých řešič pracuje, využívají metody sdružených gradientů s předpodmíněním a jeho paralelizace je založena na Schwarzových metodách a rozkladu na překrývající se podoblasti. Účinnost paralelního řešiče je vysoká díky minimálnímu množství přenesených dat a rozsahu překrytí podoblastí. Každý proces totiž komunikuje pouze se svými sousedy.

4.1.3 GEM – postprocessing

Vypočtené řešení se v programu GEM již dále nezpracovává a je řešen formou exportu dat pro program MATLAB.

4.1.4 Alternativní schéma - Systém GEM3

Systém GEM3 založený na modulové stavbě umožňuje modifikovat části softwaru. Je založen na přehlednosti dané úlohy řešení.

Souborový systém se skládá z těchto částí:

- head.txt - všeobecné informace o úloze
- tiles.txt - seznam binárních souborů
- xyz.bin - geometrie sítě (souřadnice uzlů)
- pt.bin - geometrie sítě (čtyřstěny)
- md.bin - materiály (distribuce v buňkách sítě)
- dst.bin - vektor posunutí
- fstsv.bin - tenzor napětí a deformací v elementu sítě

V systému je vypracovaná indexace dat, která umožňuje orientaci ve všech prvcích strukturované sítě (uzlech, buňkách i čtyřstěnech).

4.2 Shrnutí

Díky rozmanitosti úloh, možností algoritmů výpočtů, výpočetních modelů, formou grafických výstupů jsou programy zpravidla zaměřeny na specifickou oblast. Specializací na danou oblast ztrácejí použití nejen v jiných oblastech, ale také v částech programu zaměřených na formulování úlohy a vizualizaci výsledků. Tuto úlohu však mohou převzít programy, které jsou pro tyto účely zaměřeny. Poměrná složitost definování zadání úlohy a absence vizualizace a práce s výsledky řešiče programu GEM nahrává k využití vhodného externího programu.

5 GiD

5.1 Popis systému

GiD je produkt firmy International Center for N umerical Methods in Engineering (CIMNE) se sídlem v Barceloně, specializující se na vývoj a aplikaci numerických metod a softwaru pro tyto účely. Program je multiplatformní, vyvinutý pomocí C++, Tcl/tk a OpenGL nástroji. Byl vytvořen pro potřeby matematického modelování od preprocessingu po postprocessing. Výpočty mohou probíhat lokálně v programu, či vzdáleně v aplikaci umožňující kalkulaci z GiDu, a poté vizualizovat numerické výsledky opět v programu GiD.

5.2 Řešení problematiky ladění GiDu - preprocesing

GiD nabízí následující vlastnosti ladění [5]:

- nabídky (menu) mohou být laděny a vytvářeny pro specifické požadavky simulačního SW
- jednoduché rozhraní mezi definicí dat a simulačním SW
- jednoduché rozhraní založené na skaláru, vektoru a m atici pro vizualizaci výsledku
- nabídka (menu) pro vizualizaci výsledků může být přizpůsobena dle potřeb aplikace či analýzy

V podmínkách ladění GiD společnost zavádí pojem Problem Type.

5.2.1 Problem Type

Problem Type je kolekce souborů používaných ke konfiguraci GiDu pro jednotlivé typy analýzy. Tato kolekce zahrnuje popisující podmínky, materiály, vstupní data, jednotkový systém, symboly a formát vstupního souboru z výpočetního systému.

GiD upřednostňuje konfigurační proces uvnitř samotného programu, namísto změn ve výpočetním programu či v nezávislé utilitě.

Při tvorbě Problem Type je vytvořen podadresář se jménem a koncovkou *.gid* v adresáři *problemtypes*. Tento adresář musí obsahovat tyto konfigurační soubory:

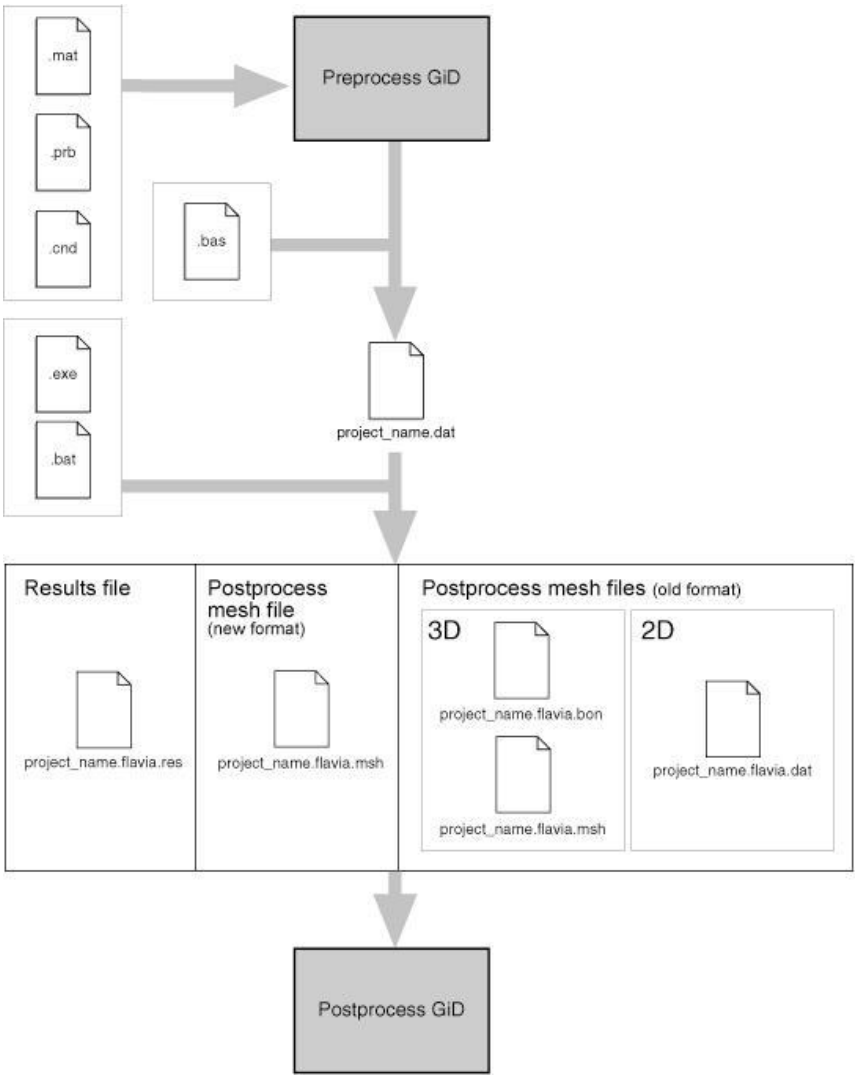
- .xml – základní konfigurace
- .cnd – definice podmínek
- .mat – vlastnosti materiálů
- .prb – data problému a intervalů (?)
- .uni – jednotkové systémy
- .sim – symboly podmínek

Dále pak tyto soubory:

- .geo – symboly geometrických definic
- .bas – informace pro přídavné soubory
- .tcl – rozšíření pro GiD psané v Tcl/Tk jazycích
- .bat – spustitelný soubor

Problem Type je sada souborů konfigurovaných výpočetním zařízením (analyzátorem) tak, že program může připravit data k analýze. Soubory pro s amotnou analýzu mají příponu

.dat (preprocesing), po analýze pak program načítá data ze souboru *.res* a *.msh* (postprocessing).



Obrázek č.1 - Diagram zobrazení systému souborů [5].

5.2.2 Předávání dat výpočetnímu modulu

Program GiD předává data výpočetnímu modulu pro výpočet v textovém souboru *.dat*. Pro definici formátu, jak má soubor *.dat* vypadat, slouží soubor *.bas*. Každý problém může mít pouze jeden konfigurační soubor. Dle nastavení souboru *.bas* lze také provádět exporty dat. Nejjednodušším exportem je výpis souřadnic uzlů (Files->Export->Using template .bas ->XYZ), případně export údajů o úloze (Files->Export->Text data report)

5.2.2.1 Konstrukce konfiguračního souboru *.bas*

Výrazy, kterým předchází hvězdička, představují příkazy.

Příklady příkazů v souboru *.bas*:

- **nelem* – vrátí celkový počet elementů v mesh síti

- **npoin* – vrátí celkový počet uzlů v mesh síti
- **loop, *end* – příkazy pro začátek a konec cyklu
- **loop nodes* – cyklus iterující uzly
- **loop elem* – cyklus iterující elementy
- **loop materials* – cyklus iterující přiřazené materiály
- **format* – příkaz pro definici formátu tisku. Po tomto příkazu musí následovat číselný formát vyjádřený syntaxí jazyka C
- **NodesNum* – vrátí identifikátor současného uzlu
- **NodesCoord* – vrátí souřadnice současného uzlu
- **NodesCoord (n, real)* – vrátí *x* (n=1), *y*(n=2), nebo *z*(n=3) souřadnici
- ...

5.2.2.2 Spouštění analýzy ve výpočetním programu

Po sestrojení všech souborů Problem type (.cnd, .mat, .prb, .sim, .bas) může být spuštěn výpočet (lze např. přímo v programu GiD). Pro tyto účely musí být nejdříve vytvořen soubor *.bat*, jehož spuštění se provádí přímo výběrem *Calculate option* v programu GiD. Pro účely výpočtu v různých operačních systémech je možno vytvořit dva soubory *.win.bat* a *.unix.bat*. Pokud se nalézá soubor *.bat* v adresáři *Problem type*, vybráním *Start* z nabídky *Calculate option* dojde ke spuštění skriptu a vygenerování souboru *.dat*.

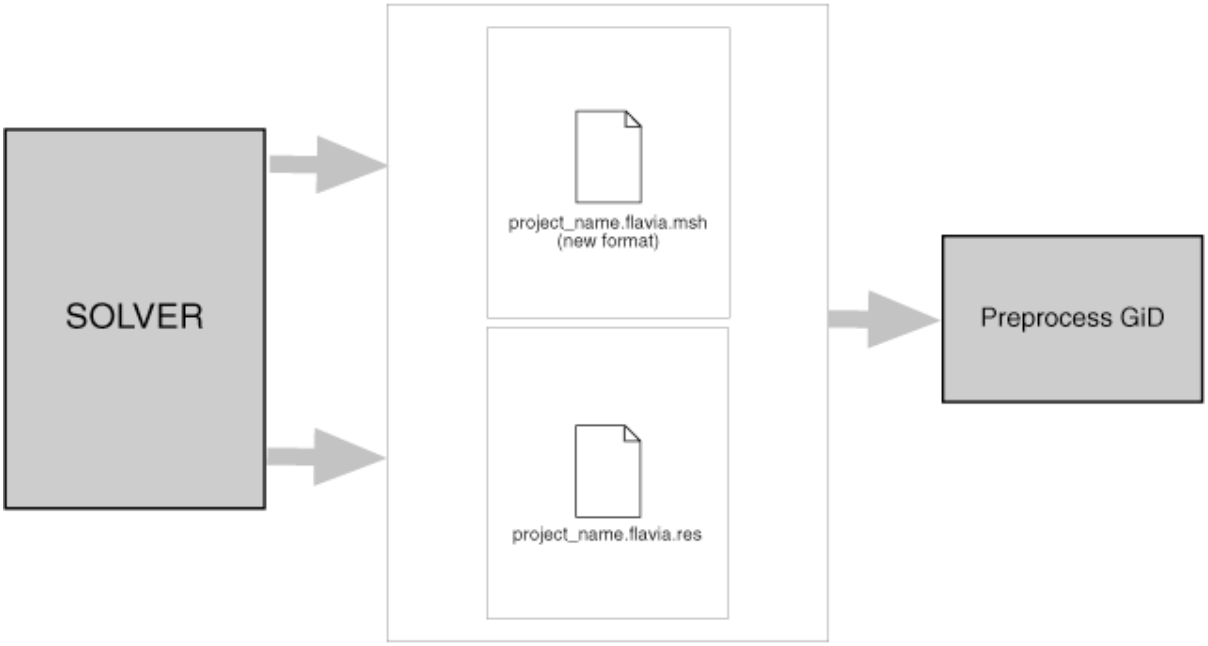
5.3 GiD - postprocesing

Program dokáže importovat značné množství modelů z různých programů. Jsou to tyto importy [4]:

- NASTRAN mesh - čte sítě z programu NASTRAN
- FEMAP - čte FEMAP Neutral ASCII soubory a binární soubory
- TECPLOT ASCII - čte TECPLOT 9.0 ASCII soubory
- 3DStudio file - čte sítě v *.3ds formátu
- XYZ points - čte nastavení bodů uložených ASCII formátu
- Cuts - čte řezy rovin, řezy sítí a řezy iso-ploch v GiDu
- Graphs - čte grafy v GiDu

Pro prezentaci modelovaného problému však potřebujeme pracovat s daty programu GEM. V tomto případě si nevystačíme jen s importem sítě.

GID v pozici postprocesu pracuje s výsledky výpočetního programu v ASCII či binárních souborech jednak pro definici sítě - Mesh Data File (*project_name.post.msh*), a také pro definici výsledků - Results Data File (*project_name.post.res*).



Obrázek č.2 - Diagram zobrazení systému souborů pro preproces.

- **ProjectName.post.msh**: (starší verze **ProjectName.flavia.msh**) – soubor musí obsahovat souřadnice uzlů a jejich spojení. Dále také může obsahovat materiál každého elementu. Pokud není zvolen žádný materiál, GiD přiřadí materiálu číslo nula.
- **ProjectName.post.res**: (starší verze **ProjectName.flavia.res**) – tento soubor musí obsahovat uzlové či gaussovy proměnné. GiD umožňuje uživateli definovat množství uzlových proměnných limitovaných pouze pamětí počítače. Pokud je volena možnost zadání gaussových proměnných, je nutné, aby soubor obsahoval definici Gaussových bodů a výsledné hodnoty na těchto bodech.

Pro verzi programu GiD 6.1.4b a vyšší nabízí software tyto nové formáty, pro které byla vyvinuta utilita **GiDpost tool** – knihovna pro jazyky C/C++ a Fortran pro vytvoření těchto postprocess souborů v ASCII formátu (od verze 6.1.4b), nebo komprimovaném binárním formátu (od verze 7.2).

5.3.1 Formát souboru „msh“ pro postproces

Tento formát souboru *.post.msh* využívají verze GiDu 6.0 a vyšší. Jak již bylo popsáno výše, slouží tento soubor k definici sítě dané úlohy řešené externě mimo program GiD. Soubor tedy obsahuje souřadnice uzlových bodů a definici elementů této sítě.

Prvním záznamem v souboru je kódování textu – jména sítě, jména výsledku a další (např. *# encoding utf-8*). Soubor začíná hlavičkou:

MESH *"mesh_name"* ***dimension*** *my_dimension* ***Elemtype*** *my_type* ***Nnode*** *my_number*

- mesh_name – jméno sítě dané úlohy
- my_dimension – 2D či 3D síť
- my_type – Point, Linear, Triange, Quadrilateral, Tetrahedra, Hexahedra či Prism – zvolený typ elementu této sítě
- my_number – počet uzlů v elementu

Dále může následovat barva sítě ve formátu # *color R G B*. Za barvou následují samotné souřadnice počínající klíčovým slovem *coordinates* a jsou ukončeny klíčovým výrazem *end coordinates*. Za souřadnicemi následuje výčet elementů začínajících klíčovým slovem *elements* a ukončeny *end elements*. Níže si ukážeme strukturu souboru sítě pro konkrétní úlohu.

```
MESH "conus" dimension 3 ElemType Hexahedra Nnode 8
Coordinates
1 0 0 0
2 0.0384612 0 0
...
115168 1.96154 2 2
115169 2 2 2
End Coordinates
Elements
1 1 2 55 54 2810 2811 2864 2863 1
2 2 3 56 55 2811 2812 2865 2864 1
...
108160 112306 112307 112360 112359 115115 115116 115169 115168 1
End Elements
```

5.3.2 Formát „res“ pro postproces

Tento formát souboru *.post.res* využívají verze GiDu 6.1.4b a vyšší. Spolu se souborem *.post.mesh* tvoří tento soubor vstupní údaje generované mimo systém GiD, které slouží pro znázornění výsledku řešení dané úlohy.

Obsah souboru je variabilní a odvíjí se od způsobu zápisu výsledků a jejich umístění. ASCII formát souboru obsahuje zpravidla na prvním řádku identifikaci souboru:

GiD Post Results File 1.0

Je zde možnost vkládat více souborů s výsledky pomocí klíčového slova *include.* Následují bloky informací o výsledcích, které vyjadřují tato klíčová slova:

- *GaussPoints* – informace o gaussových bodech – jméno, číslo gaussova bodu, souřadnice atd.
- *ResultRangesTable* – informace pro vizualizaci výsledků
- *Result* – informace o výsledku – jméno, analýza, analýza/čas kroku, typ výsledku, lokace, hodnota
- *ResultGroup* – několik výsledků v bloku – využití pro stejné hodnoty (analýzy, čas kroku a lokace)

Struktura souboru s výsledky konkrétní úlohy může vypadat následovně:

```
GiD Post Results File 1.0
Result "foot" "Napeti" 2.62242e+214 Matrix OnNodes
ComponentNames "st1" "st2" "st3" "st4" "st5" "st6"
Values
1 -1.76683 -1.76681 -2.5398 0.00369201 -0.0259932 -0.0259906
2 -1.73705 -1.73377 -2.49177 0.006136 -0.0248155 -0.0021244
```

...
64000 -0.426814 -0.426814 -0.964137 -3.50904e-06 2.85226e-07 3.03872e-07
End Values

5.3.2.1 Result Group

Výsledky mohou být shlukovány do bl oku. Každý blok má svou identifikaci v hlavičce, charakterizaci výsledků a volitelné nastavení.

Hlavička obsahuje:

- ResultGroup <<jméno analýzy>> hodnota_kroku umístění <<jméno umístění>>
  hodnota_kroku = hodnota kroku uvnitř analýzy
  umístění = kde se nalézá ResultGroup (OnNodes, OnGaussPoints – zadá se <<jméno umístění>>),

Charakterizace výsledků obsahuje:

- ResultDescription <<jméno výsledku>> typ_výsledku [: počet_komponent]
  typ_výsledku = charakteristika typu výsledku:
  - Scalar (skalár)
  - Vector (vektor)
  - Matrix (tenzor)
  - PlainDeformationMatrix
  - MainMatrix
  - LocalAxes

  počet_komponent = číslo závislé na typu výsledků. Pro konkrétní typy jsou následující:
  - 1 pro Scalar (hodnota_skaláru)
  - 3 pro Vector (X,Y,Z)
  - 6 pro Matrix (Sxx, Syy, Szz Sxy, Syz, Sxz)
  - 4 pro PlainDeformationMatrix (Sxx_hodnota, Syy, Sxy, Szz)
  - 12 pro MainMatrix (Si, Sii, Siii, ViX, ViY, ViZ, ViiX, ViiY, ViiZ, ViiiX, ViiiY, ViiiZ)
  - 3 pro LocalAxes (euler_ang_1, euler_ang_2 and euler_ang_3)

  Číslo za dvojtečkou má smysl pouze pro vektory a tenzory. Tyto hodnoty představují:
  Vector:2 : X, Y
  Vector:3 : X, Y, Z (defaultní hodnota pro vektor)
  Vector:4 : X, Y, Z, |Vector| (modul vektoru)
  Matrix:3 : Sxx, Syy, Sxy
  Matrix:6 : Sxx, Syy, Szz, Sxy, Syz, Sxz (defaultní hodnota pro tenzor)
- ResultRangesTable <<jméno RRT>>
- ComponentNames <<jméno Component 1>>, <<Name of Component 2>>
  Počet komponent (a jejich názvů) závisí na zvoleném typu. Počty názvů komponent pro příslušné typy jsou:
  1 pro Scalar Result
  3 pro Vector Result
  6 pro Matrix Result
  4 pro PlainDDeformationMatrix Result
  6 pro Main Matrix Result
  3 pro LocalAxes Result

Výsledky:

Values

location_1	result_1_component_1_value	result_1_component_2_value
	result_1_component_3_value	result_2_component_2_value
	result_2_component_2_value	result_2_component_3_value
. . .		
location_n	result_1_component_1_value	result_1_component_2_value
	result_1_component_3_value	result_2_component_2_value
	result_2_component_2_value	result_2_component_3_value

End Values

Výčet komponent pro různé typy výsledků:

1 pro Scalar results: result_number_i scalar_value

3 pro Vector results: result_number_i x_value y_value z_value

6 pro Matrix results: result_number_i Sxx_value Syy_value Szz_value Sxy_value Syz_value Sxz_value

4 pro PlainDeformationMatrix results:
result_number_i Sxx_value Syy_value Sxy_value Szz_value

12 pro MainMatrix results:
result_number_i Si_value Sii_value Siii_value Vix_value Viy_value Viz_value Viix_value Viiy_value Viiz_value Viiix_value Viiiy_value Viiiz_value

3 pro LocalAxes results:
result_number_i euler_ang_1_value euler_ang_2_value euler_ang_3_value

6 Postprocessing

Pro předávání dat, která jsou potřebná pro vizualizaci v programu GiD je potřeba rozdělit problém do dvou dílčích částí, a to část předání dat reprezentujících znázornění sítě (uzlů a elementů) a část předávání dat pro znázornění výsledků řešení dané úlohy. K sestavování dvou souborů, s nimiž GiD pracuje, byl firmou CIMNE International Center for Numerical Methods in Engineering (Barcelona) vyvinut nástroj Gidpost. Jedná se o soubor funkcí (knihovnu) pomocí kterého lze předávat výsledky GiDu v ASCII či binárním formátu. Tato knihovna byla vyvinuta pro dvě skupiny jazyků, a to jazyky C/C++ a jazyk FORTRAN. Jelikož program GEM byl vyvinut pomocí jazyka FORTRAN, byl právě tento jazyk zvolen pro vývoj rozhraní mezi softwarem GEM a softwarem GiD.

Cílem postprocessingu, využívající program GiD, je znázornit síť úlohy, vyznačit v ní změnu materiálu a vykreslit výsledky metody konečných prvků. Výsledky této metody jsou interpretovány v podobě tenzorů napětí a deformací řešené úlohy.

Pro účely testování posprocessingu jsou použity tři úlohy řešené systémem GEM. První úlohou je jednoduchá testovací úloha s označením D0 (27 uzlů sítě). Sloužila k odladění programu rozhraní při jeho počátcích, neboť objemem dat nevyžaduje plnou verzi programu GiD. Druhou testovací úlohou je úloha FOOT (68921 uzlů sítě). Poslední testovací úloha má označení CONUS (115169 uzlů sítě) pro řešení sítě čtyřstěnů.

6.1 Znázornění sítě

6.1.1 Výběr funkcí pro znázornění sítě

Jak již bylo zmíněno v kapitole 5.3.1 popisující program GiD, je potřebné vytvořit soubor **.post.mesh*, který bude v programu GiD zobrazovat rozmístění uzlů sítě a jejich vzájemné propojení. Nejprve založíme soubor pro vkládání dat o síti pomocí funkce knihovny gidpost:

```
int GiD_OpenPostMeshFile(char* FileName,GiD_PostMode Mode);
```

Pro naformátování bloku dat pro znázornění sítě slouží tato funkce:

```
int GiD_BeginMesh(char* MeshName,GiD_Dimension Dim,GiD_ElementType EType, int NNode);
```

Její popis je uveden v kapitole 5.3.1. Do tohoto bloku voláme funkce pro ukládání dat o uzlech sítě:

```
int GiD_BeginCoordinates();          - začátek bloku vkládání uzlů
int GiD_WriteCoordinates(int id,double x,double y,double z);
                                     - vkládání souřadnic uzlů
int GiD_EndCoordinates();            - ukončení bloku vkládání uzlů
```

a pro ukládání dat o elementech sítě:

```
int GiD_BeginElements();             - začátek bloku vkládání elementů
int GiD_WriteElement(int id,int nid[]);
                                     - vkládání elementů, tedy propojení uzlů
```

int GiD_WriteElementMat(int id,int nid[]);
- alternativa vkládání (poslední hodnota reprezentuje číslo materiálu)

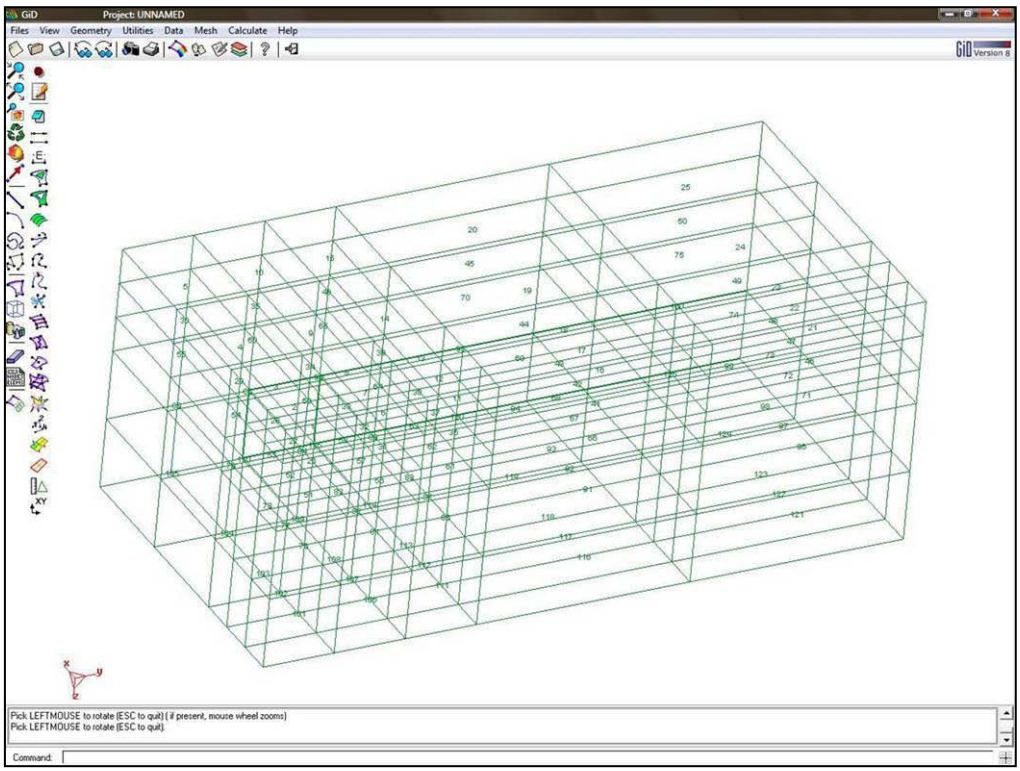
int GiD_EndElements(); - ukonční bloku vkládání elementů

Celý blok ukončují funkce pro uzavření zápisu sítě a souboru se sítí:

int GiD_EndMesh();
int GiD_ClosePostMeshFile();

6.1.2 Výběr dat pro znázornění sítě

Zdroj dat pro tuto úlohu lze nalézt jednak v souboru *fv.gem*, odkud lze načítat počet uzlů a jejich mezních hodnot, a d ále v souboru *fx.gem*, který obsahuje přímo údaje o souřadnicích uzlů diskretizační sítě. Výsledkem jednoduché úlohy může být výstup znázorněný na Obr.č.3.



Obrázek č.3 – Znázornění sítě v programu GiD.

Při sestavování souboru **.post.mesh* lze definovat mimo umístění bodů také typ elementů a barvu sítě.

6.1.3 Shlukování sítí

Některé úlohy, například znázornění hexagonální a tetragonální sítě, nebo znázornění posunutí, vyžadují vykreslení více sítí. Třída umožňující předat programu GiD více než jednu síť se jmenuje GID_BEGINMESHGROUP resp. GID_ENDMESHGROUP. Obalením kódu programu, ve kterém jsou zapisovány uzly a elementy sítí, těmito párovými funkcemi dojde k zápisu těchto sítí do souboru **.post.mesh*.

6.2 Znázornění výsledků

6.2.1 Výběr funkcí pro znázornění výsledků

V kapitole 5.3.2 bylo uvedeno, že pro předávání výsledků pro zobrazování v programu GiD slouží soubor **.post.res*. Rozhodující při tvorbě tohoto souboru je volba typů výsledků. Výsledky naší vzorové úlohy jsou hodnoty napětí a deformací v podobě tenzorů. Pro náš případ tedy byla zvolena funkce knihovny gidpost *GiD_Begin3DMatResult*, neboť právě tato funkce slouží pro zápis výsledků v podobě tenzorů, které jsou znázorněny v 3D síti. Její popis notace spolu se vstupními parametry je následující:

```
int GiD_Begin3DMatResult(char* Result,char* Analysis,float step,
                         GiD_ResultLocation Where,
                         char* GaussPointsName,char* RangeTable,
                         char* Comp1,char* Comp2,char* Comp3,
                         char* Comp4,char* Comp5,char* Comp6);
```

kde:

- char* Result - název sady výsledků
- char* Analysis - název analýzy
- float step - krok
- GiD_ResultLocation Where - umístění výsledků: 0 = na uzlech sítě 1 = na Gaussových bodech
- char* RangeTable - jméno RangeTable, nebo NULL
- char* Comp1 – Comp6 - jména složek tenzoru

Podoba funkce knihovny gidpost pro náš konkrétní případ může vypadat takto:

Nejprve založíme soubor pro vkládání výsledků:

```
CALL GID_OPENPOSTRESULTFILE('test.post.res',2)
```

Poté naformátujeme blok vkládání výsledků dle výše popsané notace:

```
CALL GiD_Begin3DMatResult('Result','Napeti',1.0,0,NULL,NULL,
.'st1','st2','st3','st4','st5','st6');
```

V následujícím kódu můžeme vkládat blok výsledků pro znázornění v GiDu. Pro náš případ využijeme funkci pro vkládání tenzorů v 3D síti. Její notace je následující:

```
int GiD_Write3DMatrix(int id,double Sxx,double Syy,double Szz, double Sxy,double Syz,double Sxz);
```

kde:

- int id - reprezentuje jedinečnou identifikaci
- double S.. - hodnoty složek tenzoru

V našem konkrétním případě může vypadat funkce pro vkládání výsledků takto:

```
CALL GID_WRITE3DMATRIX(i,st1,st2,st3,st4,st5,st6)
```

6.2.2 Výběr dat pro znázornění výsledků

Výsledky vzorové úlohy v programu GEM v části STRESS generují tenzory napětí (ST) a deformací (DF), které jsou binárně uloženy do souboru *fsts.g32*. Protože jsou tyto tenzory pro jemnější síť čtyřstěnů, je nutné přihlédnout k tomuto faktu při tvorbě sítě.

6.2.3 Postup

Jak již bylo uvedeno, program využívá knihovny *gidpost* pro využívání funkcí generujících soubory pro import v GiDu. Z toho důvodu je nutné buďto při kompilaci uvést cestu ke knihovně nebo program nakopírovat do adresáře */gidpost/unix/debug/*. Do tohoto adresáře je nutné nakopírovat také výstupní soubory programu GEM pro danou úlohu, ze kterého se čerpají data pro následnou práci v GiDu. Jsou to soubory *fv.g32* – pro definici úlohy, *fx.g32* – pro načítání souřadnic sítě, *fel.g32* – pro načítání informací o buňce a materiálu, *fu.g32, fsts.g32, fstsv.g32* – pro načítání hodnoty výsledků.

Příkaz pro překlad programu v operačním systému Linux je následující:

gfortran -o exe GemToGid.f libgidpostd.a -lstdc++.

Po vykonání programu spustitelným souborem dojde k vytvoření dvou souborů reprezentující síť (**.post.msh*) a výsledky (**.post.res*). Názvy souborů začínají zvolenou možností sítě čtyřstěnů (C), nebo sítě šestistěnů (S). Dále v názvu následuje prvních deset znaků názvu úlohy načtené ze souboru *fv.g32*. Soubory lze importovat v režimu *Postprocess* v programu GiD (viz kapitola 6.2.4).

Samotný program rozhraní obsahu metody *ctyrsteny()* a *sestisteny*() pro vytvoření importů GiDu a dále dvě pomocné metody *read_fv()* pro načtení informací o úloze a *alokace()* pro vyřešení absence alokace dynamických polí ve Fortranu77.

6.2.4 Vizualizace výsledků v programu GiD

Program GiD nabízí pro práci dva režimy. První z nich se jmenuje *Preprocess*, jehož hlavní úlohou je namodelovat úlohu zadáním vstupních údajů. Ty lze zadat buďto ručně, nebo importovat různé formáty externích softwarů. Mezi podporovanými soubory je také soubor *.mesh, tedy ASCII formát GiDu pro definici sítě. V tomto módu lze síť různě modifikovat.

Druhým módem programu GiD je režim *Postprocess*, ve kterém GiD umožňuje zobrazovat výsledky řešených úloh a pracovat s nimi. Do tohoto módu se lze přepnout buďto pomocí ikony pro přechod mezi stavy, nebo prostřednictvím menu Files -> Postprocess. Zde program umožní rovněž načtení různých formátů externích softwarů (viz. kapitola 5.3), ale také soubory **.mesh* a **.res*.

Program GiD umožňuje tyto typy zobrazení načtených výsledků:

- Contour fill - zobrazování barevných zón, ve kterých se pohybuje proměnná mezi dvěma zadanými hodnotami. U zobrazení tenzoru lze volit ze Sxx, Syy, Szz, Sxy, Syz a Sxz složek, dále pak ze složek Si, Sii, Siii.
- Smooth contour fill - obdoba Contour fill pro zadání s gaussovými body
- Contour lines - obdoba Contour fill, ale vykreslení isočar pro konkrétní hodnoty proměnné v uzlech
- Contour Ranges - totožné zobrazení s Contour fill, avšak barevné zóny jsou tvořeny specifikací v „Result range table“
- Show Min and Max - zobrazení minima a maxima výsledku
- Display vectors - zobrazení vektorů a tenzorů. U kladného tenzoru je barva červená, u záporného modrá. U zobrazení tenzoru lze volit ze Sxx, Syy, Szz, Sxy, Syz a Sxz složek, dále pak ze složek Si, Sii, Siii.
- Iso surface - vykreslení ploch fixních hodnot uvnitř objemu sítě, pro povrch sítě je vykreslena čára. Možnosti tohoto zobrazení jsou:
    - Exact - po vložení několika hodnot jsou zobrazeny isoplochy pro tyto hodnoty

    - Automatic - po zadání počtu isoploch GiD vypočítá hodnoty mezi minimem a maximem a zobrazí je
    - Automatic Width - po zadání „width“ vytvoří GiD tolik isoploch, kolik je potřeba mezi definovaným minimem a maximem
- Stream Lines - zobrazení „stream line“, „fluid dynamics“ a „particle tracing“ ve vektorovém poli. GiD se dotáže na bod, od kterého začne vykreslovat „steam line“. Tento bod lze zadat výběrem uzlu v rovině řezu sítí.
- Graphs - zvolení a mazání grafů pro výsledky definované v uzlech
- Graph Lines description - vykreslování grafů (bližší pohledy na výsledky)
- Result surface - zobrazuje 3D plochy nad sítí pomocí normál.
- Deform Mesh - umožňuje deformovat objemy, povrchy a řezy pomocí uzlových vektorů a faktorů. GiD vykreslí tyto deformované objekty
- Line diagram - tato možnost je zpřístupněna, pokud jsou v síti použity „line elements“. Zobrazován je buďto „Scalar Diagram“ – vykreslení v řezu paralelnímu k obrazovce, nebo „Vector Diagram“ – vykreslení v řezu obsahujícím výsledky vektoru a vektor, který čáru definuje
-

Program GiD vypočítá složky Si, Sii a Siii, které reprezentují vlastní čísla a vektory maticových výsledků.

6.2.5 Nastavení některých zobrazení

6.2.5.1 Styly zobrazení

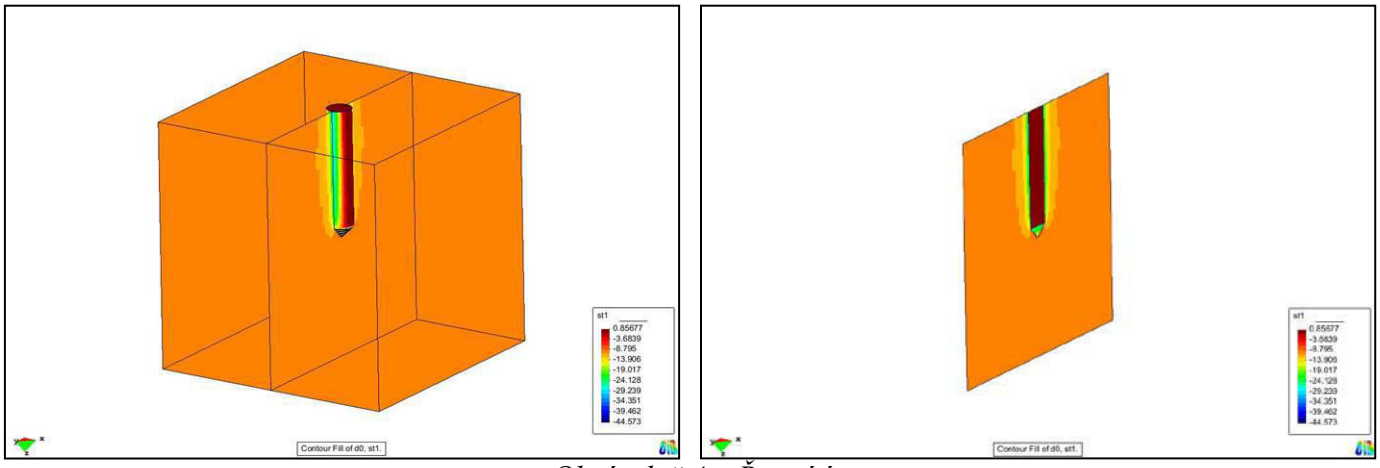
Aby byly výsledky, které chceme zobrazit prostřednictvím programu GiD, dobře čitelné, umožňuje program v režimu *Postprocess* několik možností zobrazení. Funkce je pojmenována *Dispay Style* a umožňuje nastavovat režimy zobrazení všech vrstev v úloze. Volbou nastavení je barva objektu, jeho průhlednost, viditelnost a styl zobrazení. Stylů zobrazení je celkem devět – Hidden Bound, All Lines, Hidden Lines, Body, Body Bound, Body Lines, Boundaries, Points a Poinst Bound. Kombinací těchto stylů spolu s funkcí *Culling* (redukce) dává program možnost zobrazit síť a výsledky ve všech podobách, které jsou pro jejich prezentaci potřeba.

6.2.5.2 Řezy

Pro znázornění výsledků uvnitř úlohy slouží nástroj Cut (řez). GiD umožňuje provádět řez jednak objemem těles (výsledků), ale také řezy povrchu či řezy jiných řezů. Výsledkem řezu je takový počet nových objektů, kolika materiály řez prochází.

Řez lze definovat třemi způsoby:

- 2 points - volbou dvou bodů je proveden řez sítí a vytvořená rovina je kolmá k pohledu zobrazení (viz Obr.č.4).

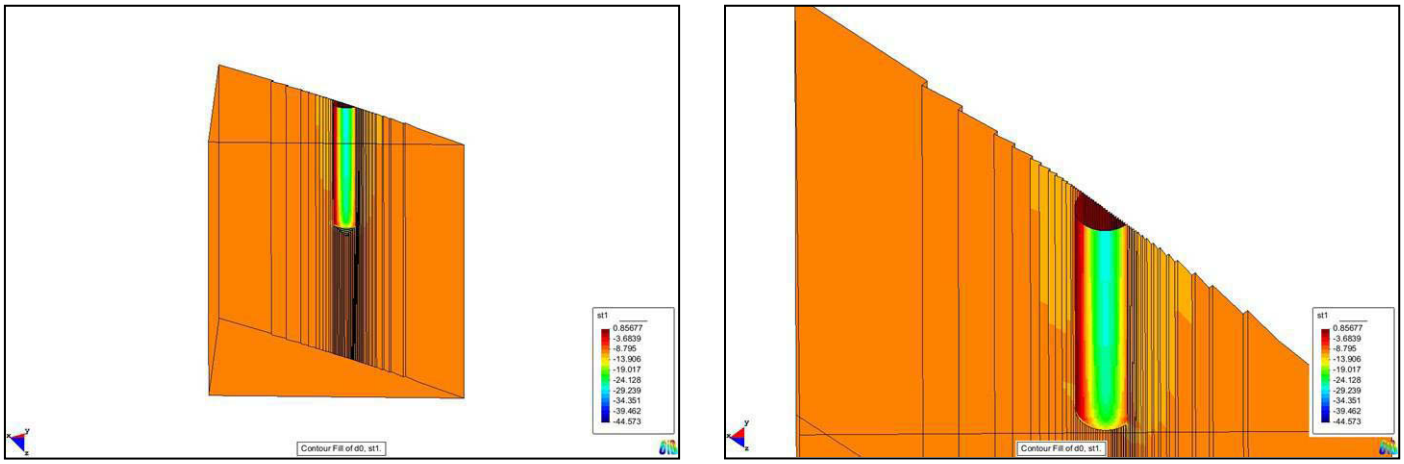


Obrázek č.4 – Řez sítí.

- 3 points - rovina řezu je definována výběrem tří bodů. Při této metodě řezu je důležitým faktorem jednoznačnost výběru bodu, kterým rovina prochází.
- Succesion - výsledkem je zadaný počet rovin řezu ve vybrané oblasti (viz Obr.č.5).

Obrázek č.5 – Posloupný řez.

Další možností provedení řezu je metoda *Divide volume sets*, která vede rovinu řezu bez toho, aby byly řezány vlastní elementy. Příklad takového řezu je uveden na Obr.č.6.



Obrázek č.6 – Řez dělením sítě.

6.2.5.3 Deformace

Deformace objektů, ploch a řezů jsou umožněny pouze pro výsledky definované v uzlech sítě. V nabídce *View Results & Deformation* jsou pro tyto účely záložky *Main Mesh* a *Reference mesh*. Nastavením hodnoty *factor* umožní program vykreslit deformace s násobícím faktorem, který zviditelní mnohdy okem nepozorovatelné výsledky.

6.2.5.4 Animace

Dalším nástrojem pro práci s výsledky je možnost animace. Animovat lze buďto časovou závislost sady výsledků (identifikovaných pomocí předem zadané hodnoty *step* u výsledků), případně animace deformací. Výstupním formátem může být soubor TIFF, MPEG nebo AVI.

6.2.5.5 Práce s makry

Pro usnadnění práce s výsledky, která se v některých případech skládá z opakující se sady stejných příkazů, má program GiD k dispozici možnost vytvořit záznam makra. Záznam lze poté upravovat, nastavovat mu klávesové zkratky, zobrazování na liště aj.

7 Závěr

Cílem této práce bylo pochopení systému GEM, seznámení se se současným stavem preprocessingu a postrpocessingu tohoto programu. Dále bylo potřeba nastudování softwaru GiD a jeho práci s výstupy externích programů. Stěžejním pilířem této práce bylo navrhnutí rozhraní pro práci v GiDu s importem výsledků programu GEM.

Ačkoli se během zpracovávání objevily překážky spočívající v podpory GiDu některých funkčních prvků (řezů v Gaussových bodech, načítání kroků, deformací), stabilitě programu GiD při práci s větším objemem dat, případně práci se strukturovanou sítí, bylo možno v práci pokračovat po aktualizaci GiDu z verze 9.0.6 O fficial na verzi 9.3.1.b Developer. Tato verze se ukázala stabilnější, a také zpřístupnila možnost použití řezů v Gaussových bodech.

Největším problémem při vypracování bylo použití všech knihoven rozšiřujících Fortran pro účely vytvoření souborů definujících síť a výsledky řešené úlohy pro práci v GiDu.

I přes všechny překážky se podařilo dovést práci ke zdárnému konci a vytvořit rozhraní využívající data konkrétních úloh systému GEM, která jsou předávána programu GiD. Tyto výsledky mohou být zobrazeny buďto v síti reprezentovanou šestistěny pro rychlejší práci při vizualizaci, případně v síti reprezentovanou čtyřstěny pro přesnější vykreslení komplikovaných oblastí úlohy.

Díky této práci a závěrům, které předkládá, je možno zefektivnit proces matematického modelování převážně v oblasti postprocessingu a také nastínila možnost využití definování úlohy a převzetí roli preprocesoru.

Dalším rozšířením této práce je, při možnosti aktualizace programu GiD na verzi 11, využití veškerého potenciálu programu GiD. Jedná se především o rychlejší vizualizaci (s využitím potenciálu moderních grafických karet), rozšiřující funkce postprocessingu, výpočty skalárních charakteristik výsledků a jejich animace atd.

8 Použitá literatura

[1] JAKL, O., BLAHETA, R., S TARÝ, J. *Library of parallel PCG solvers for problems in geomechanics*. In ALGORITMY 2000. Ed. A. Handlovičová, M. Komorníková, K. Mikula, D. Ševčovič. Bratislava: Slovak University of Technology. 2000. 11, 388-398.

[2] MÍKA, S., PŘIKRYL, P., BRANDNER, M. *Speciální numerické metody.* 1. vyd. Plzeň : Vydavatelský servis, 2006. 305 s.

[3] KOLÁŘ, V., NĚMEC, I., KANICKÝ, V. *FEM principy a pr axe metody konečných prvků*. Praha: Computer Press, 2001. 401 s.

[4] GiD. *Reference manual* [online].GiD, © 2012 [cit. 2012-08-13]. Dostupné z: http://www.gidhome.com/support/manuals

[5] GiD. *Customization manual* [online].GiD, © 2012 [cit. 2012-08-13]. Dostupné z: http://www.gidhome.com/support/manuals

[6] GiD. *GiD Conference 2008 Cour se* [online].GiD, © 2012 [cit. 2012-08-13]. Dostupné z: http://www.gidhome.com/support/manuals

[7] Ústav geoniky AV ČR, v.v.i. Konečněprvkový software GEM. *Ugn.cas.cz* [online]. © 2000 - 2012 [cit. 2012-08-13]. Dostupné z: http://www.ugn.cas.cz/index.php?l=cz&a=&p=other/sw-gem/gem.php

[8] Blaheta, R. - Kohut, R. *Benchmarks and programs for testing iterative solvers for 3D problems of geomechanics*. Ostrava: Institute of Geonics Academy of Sciences of the Czech Republic 1995. 25 s.

Seznam obrázků

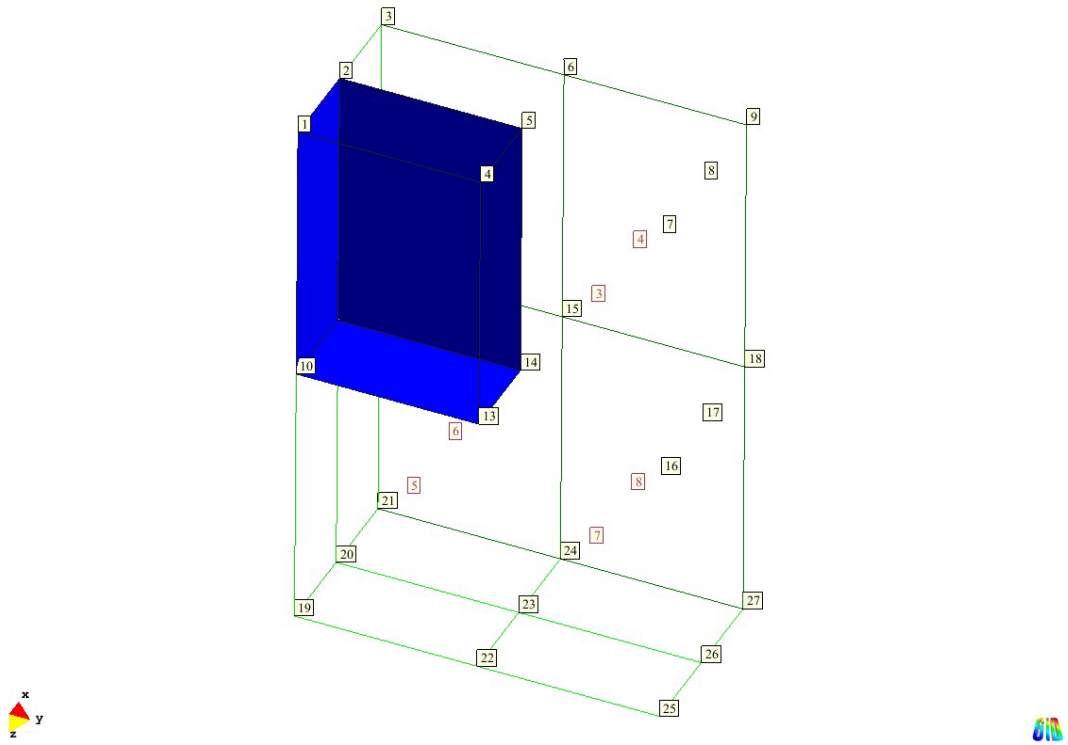
Obrázek č.	Název	Strana
1	Diagram zobrazení systému souborů.	14
2	Diagram zobrazení systému souborů pro preproces.	16
3	Znázornění sítě v programu GiD.	21
4	Řez sítí.	25
5	Posloupný řez.	25
6	Řez dělením sítě.	25

Seznam příloh

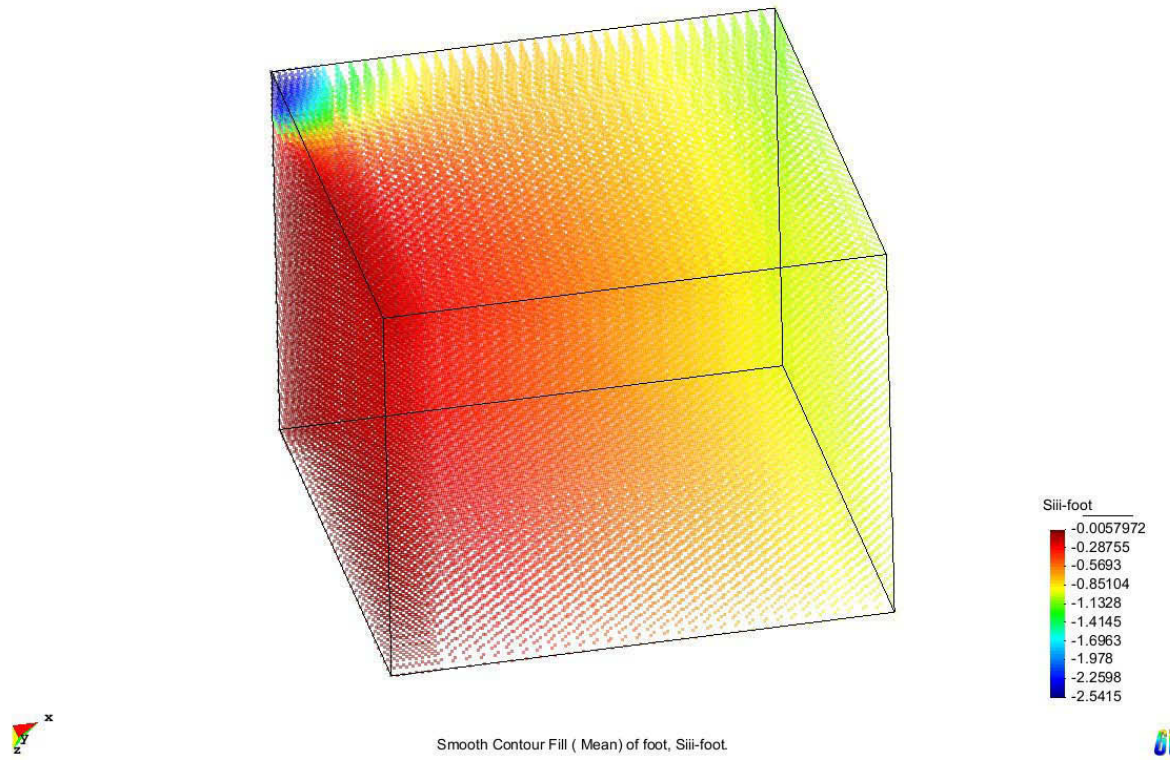
Příloha č.	Název
1	D0-síť.
2	FOOT-Contour Fill.
3	CONUS-výsledky napětí.
4	CONUS-detail sítě čtyřstěnů.
5	Výpis části programu rozhraní.

9 Přílohy

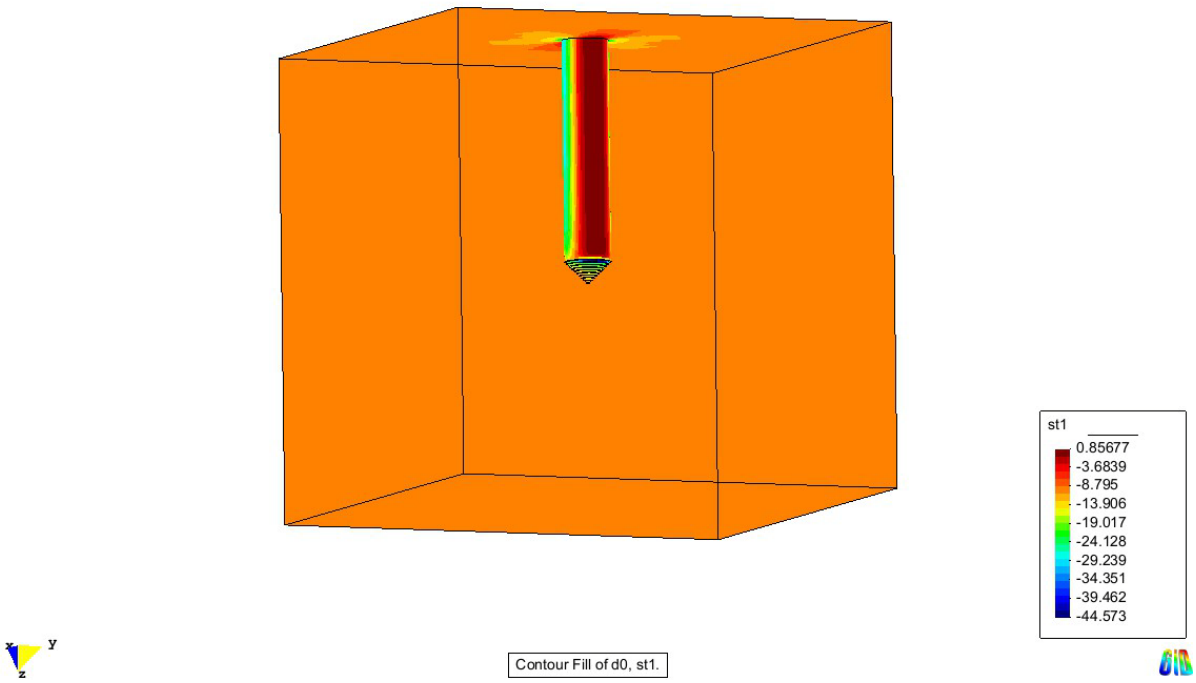
Příloha č.1 – D0-síť.



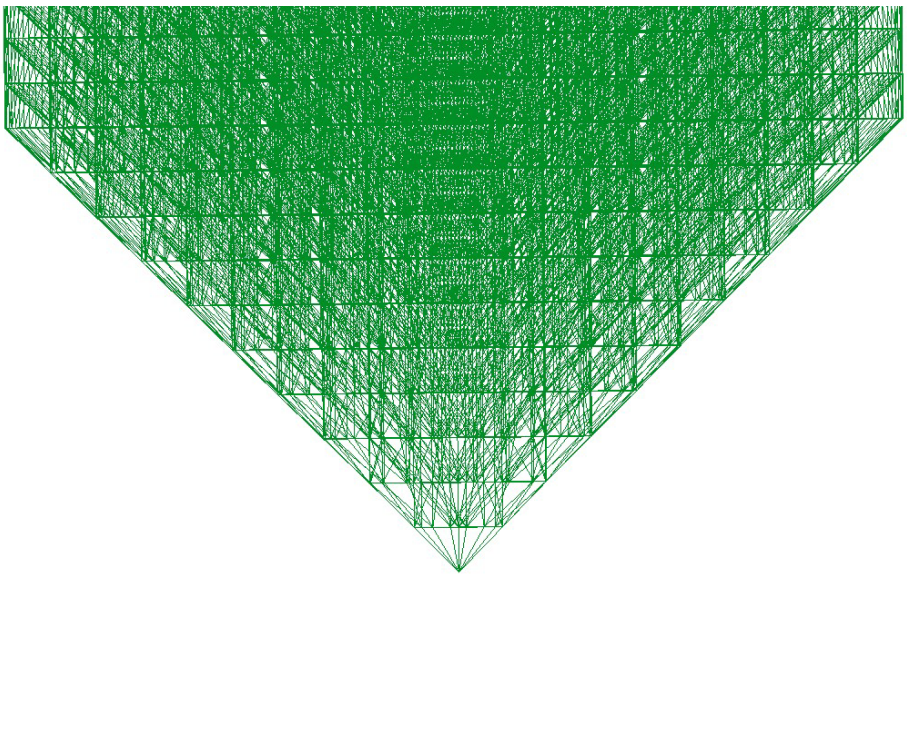
Příloha č.2 – FOOT-Contour Fill.



Příloha č.3 – CONUS-výsledky napětí.



Příloha č.4 – CONUS-detail sítě čtyřstěnů.



Příloha č.5 – Výpis části programu rozhraní.

```
      PROGRAM GemToGid
      implicit none
!     preklad:
!     gfortran -o exe vektor.f libgidpostd.a -lstdc++

      CHARACTER   task_name*50, date*8, n_res*60, volba*1
      CHARACTER*4 NULL

      INTEGER          nn, nt, nx, ny, nz, mat, i, j, k
      INTEGER          space !pozice mezery v nazvu
      INTEGER*2   nmat, hbc
      INTEGER*4   maxn, idx, x, y, z, nid(1:9), nid_h(1:5)
      INTEGER*4   it(4), iv
      INTEGER*4   nd, nc, nel, nc_max

      PARAMETER( nc_max = 1000000 )
      PARAMETER( maxn = 140 )
      INTEGER          elems(1:nc_max,1:8)

      REAL*4  st(6), df(6)
      REAL*4  xx(maxn,maxn,maxn), yy(maxn,maxn,maxn), zz(maxn,maxn,maxn)
      REAL*8  st1, st2, st3, st4, st5, st6
      REAL*8  nodes(1:nc_max,1:3)
      REAL*8  rx, ry, rz

      CALL read_fv(nn)

      CALL alokace(nc,nx,ny,nz,nn)

      WRITE (*,*) 'Byla nactena uloha'
      WRITE (*,*) task_name
      WRITE (*,*) '------------------'
1     WRITE (*,*) 'Zadejte prosim typ site.'
      WRITE (*,*) ' c[ctyrsteny], s[sestisteny]: '
      READ  (*,*)  volba

      IF (volba .EQ. 'c') THEN
        WRITE (*,*) 'Zadali jste: ctyrsteny. '
        CALL ctyrsteny()
      ELSE IF (volba .EQ. 's') THEN
        WRITE (*,*) 'Zadali jste: sestisteny. '
        CALL sestisteny()
      ELSE
        WRITE (*,*) 'Spatne zadani!'
        GOTO 1
      END IF


      NULL = CHAR(0)//CHAR(0)//CHAR(0)//CHAR(0)


      CONTAINS
      …
```